Radically Open Security

Penetration Test Report

Horizontal

Tella mobile apps

V 1.0
Amsterdam, June 25th, 2025
Public

Document Properties

Client	Horizontal
Title	Penetration Test Report
Targets	Tella Android 2.7.1 Tella Android FOSS 2.0.15 Tella iOS
Version	1.0
Pentesters	Tim Hummel, Niko Schmidt, Abhinav Mishra
Authors	Tim Hummel, Niko Schmidt, Abhinav Mishra, Marcus Bointon
Reviewed by	Marcus Bointon
Approved by	Melanie Rieback

Version control

Version	Date	Author	Description
0.1	May 9th, 2024	Tim Hummel, Niko Schmidt, Abhinav Mishra	Initial draft
0.2	August 30th, 2024	Marcus Bointon	Review
1.0	June 25th, 2025	Marcus Bointon	1.0

Contact

For more information about this document and its contents please contact Radically Open Security B.V.

Name	Melanie Rieback
Address	Science Park 608 1098 XH Amsterdam The Netherlands
Phone	+31 (0)20 2621 255
Email	info@radicallyopensecurity.com

Table of Contents

1 Executive Summary

1.1 Introduction

Between May 6, 2024 and August 23, 2024, Radically Open Security B.V. carried out a penetration test for Horizontal.

This report contains our findings as well as detailed explanations of exactly how ROS performed the penetration test.

1.2 Scope of work

The scope of the penetration test was limited to the following targets:

- Tella Android 2.7.1
- Tella Android FOSS 2.0.15
- Tella iOS

1.3 Project objectives

ROS will perform a penetration test and code review of Tella Android, FOSS and iOS applications with Horizontal in order to assess their security. To do so ROS will access their respective git repos, and guide Horizontal in attempting to find vulnerabilities, exploiting any such found to try and gain further access and elevated privileges.

1.4 Timeline

The security audit took place between May 6, 2024 and August 23, 2024.

1.5 Results In A Nutshell

During this crystal-box penetration test we found 2 Moderate, 3 Low and 1 Unknown-severity issues.

Android

We took Subgraph's public previous audit report as a starting point, and performed an analysis on the differences between the version that was previously tested, and the current state of the projects. Readers that want to learn more about Tella's inner workings are encouraged to read Subgraph's report as it explains the technical details well; their report can be downloaded from Tella here: https://tella-app.org/security-and-privacy/.

We took version 2.0.5, as referenced by Subgraph, and compared it to the more recent 2.7.1 version, checking the impact of new features and code changes since the Subgraph audit, though we did also check the cryptographic algorithms used in the light of intervening developments. We also checked the Android FOSS version 2.0.15.

The Android app's manifest permits unencrypted communications HRZ-010 (page 13). Both Tella Android HRZ-007 (page 14) and Tella Android FOSS non-finding NF-001 (page 16) use outdated, vulnerable dependencies that should be updated. We did not identify any other major flaws, though we do recommend several improvements in the cryptographic algorithms HRZ-009 (page 12) and repository maintenance on GitHub HRZ-006 (page 11).

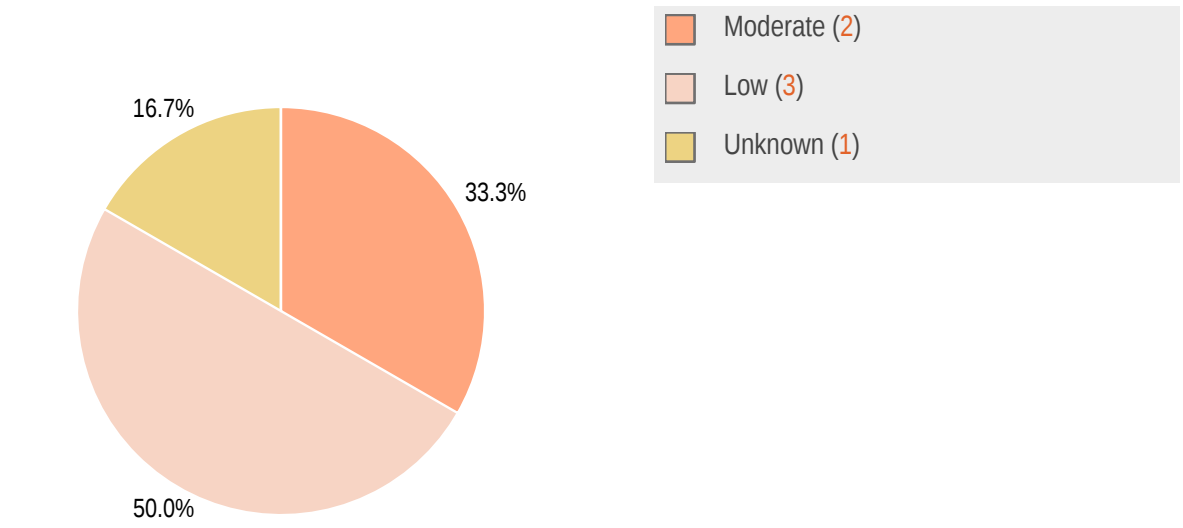
iOS

In Tella for iOS, changing security settings does not require a password HRZ-018 (page 8). We have several suggestions about how to improve the security of the WebView HRZ-020 (page 10).

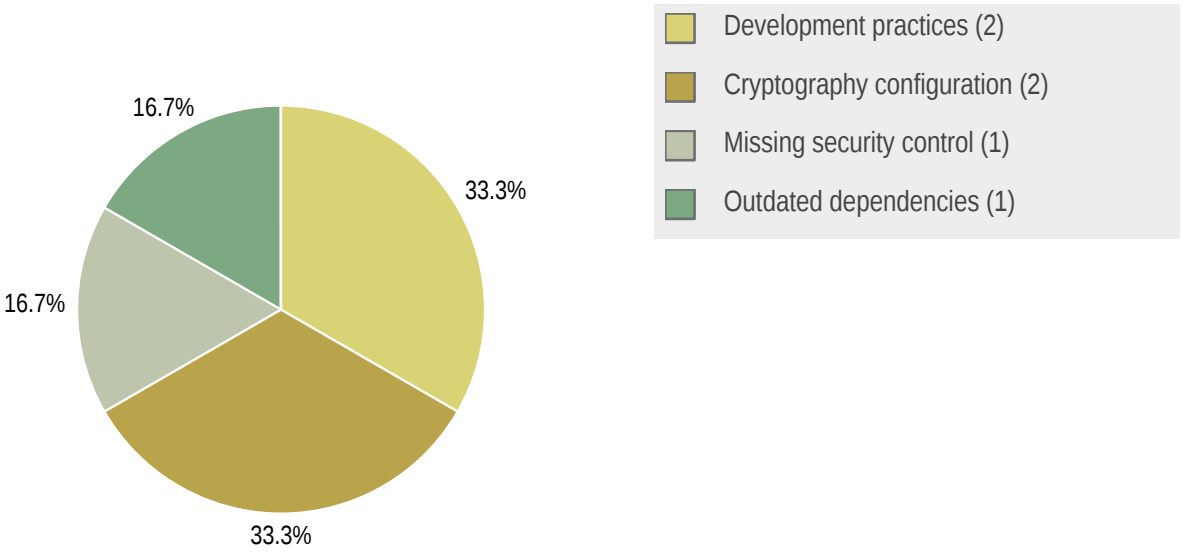
1.6 Summary of Findings

ID	Type	Description	Threat level
HRZ-018	Missing security control	The mobile app enables users to modify security settings, such as adjusting the lock timeout and enabling deletion after failed unlock attempts. However, unlike changing a PIN or password, users are not required to re-authenticate themselves before making these changes.	Moderate
HRZ-020	Development practices	The iOS app has a webview implementation at Tella-iOS/Tella/Scenes/HomeView/Views/FileDetailView/WebView.swift, however this implementation can be improved by adding some more security checks and controls.	Moderate
HRZ-006	Development practices	The branch structure, release tags, and readme offered on GitHub are confusing and outdated, making it hard to identify and install the most up-to-date version.	Low
HRZ-009	Cryptography configuration	PBKDF2 has limited brute-force resistance against GPUs, and iteration counts used are below recommendations.	Low
HRZ-010	Cryptography configuration	The app manifest explicitly disables a protection mechanism against clear-text traffic.	Low
HRZ-007	Outdated dependencies	Tella Android uses dozens of outdated or deprecated dependencies, some of which contain known vulnerabilities.	Unknown

1.6.1 Findings by Threat Level



1.6.2 Findings by Type



1.7 Summary of Recommendations

ID	Type	Recommendation
HRZ-018	Missing security control	• Require reauthentication before allowing any change in the application's security settings.
HRZ-020	Development practices	• Disable Javascript. • Validate URLs and enforce HTTPS. • Avoid excessive caching.
HRZ-006	Development practices	• Keep the main branches up to date. Update the links in the READMEs. Keep the releases up to date.
HRZ-009	Cryptography configuration	• Increase PBKDF2 iteration counts. • Change the key derivation algorithm to one that's more brute-force resistant (such as Argon2id).
HRZ-010	Cryptography configuration	• Do not allow clear-text traffic in Android manifest.
HRZ-007	Outdated dependencies	• Update outdated dependencies. • Check and replace deprecated dependencies.

2 Findings

We have identified the following issues:

2.1 HRZ-018 — Mobile App: Changing security settings does not require re-authentication

Vulnerability ID: HRZ-018 **Status:** New

Vulnerability type: Missing security control

Threat level: Moderate

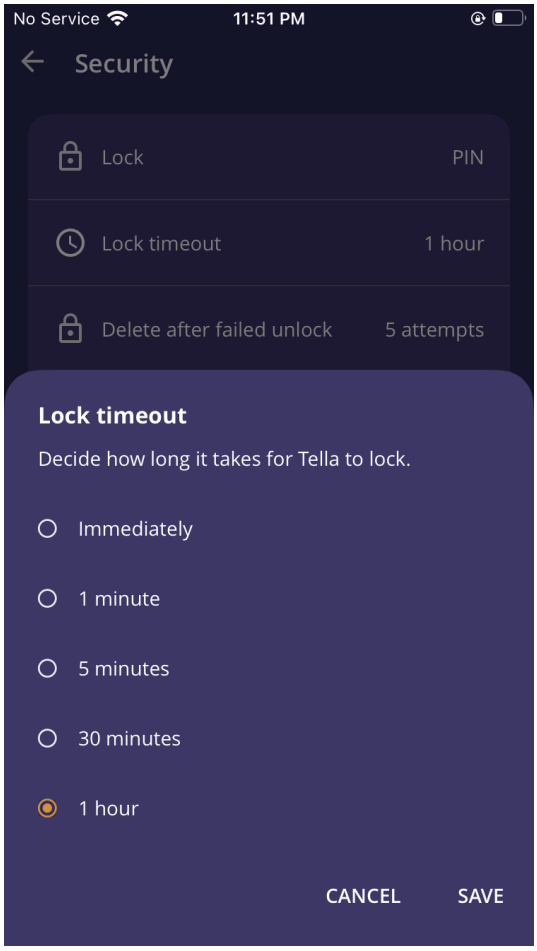
Description:

The mobile app enables users to modify security settings, such as adjusting the lock timeout and enabling deletion after failed unlock attempts. However, unlike changing a PIN or password, users are not required to re-authenticate themselves before making these changes.

Technical description:

Security settings are a critical component of the application. Consider a scenario where a user has set the lock timeout to "Immediately." If someone gains temporary access to the unlocked app, they could change this setting to "1 hour" and disable the "Delete after failed unlock" option. This would enable an attacker to perform a brute-force attack on the PIN without the risk of the app deleting data after failed attempts and without the app locking itself for an hour (with physical access).

Modifying security settings



Impact:

A malicious user can modify the app's security settings without having access to the PIN.

Recommendation:

- Require reauthentication before allowing any change in the application's security settings.

2.2 HRZ-020 — iOS: Improvements to webview implementation

Vulnerability ID: HRZ-020	**Status:** Resolved
Vulnerability type: Development practices	
Threat level: Moderate	

Description:

The iOS app has a webview implementation at `Tella-iOS/Tella/Scenes/HomeView/Views/FileDetailView/WebView.swift`, however this implementation can be improved by adding some more security checks and controls.

Technical description:

We recommend several changes to improve the Webview's security:

Disable Javascript

Disabling JavaScript in the WebView can enhance security, especially if you do not need JavaScript for your application's functionality. JavaScript can be a vector for various types of attacks, such as cross-site scripting (XSS), which can compromise the security and integrity of your app.

URL Validation

Directly converting a string to a URL using `URL(string: url)!` without any validation can lead to various attacks if the input URL is not trusted or properly sanitized.

Cache Policy

Using cachePolicy: `.returnCacheDataElseLoad` might cause outdated content to be loaded. A more secure approach would be to use `.reloadIgnoringLocalCacheData` to ensure fresh content is always loaded.

Update :

Retest Result

As per the changes done in `https://github.com/Horizontal-org/Tella-iOS/pull/201/commits/af3687483d8de828d60b5b23b6ff8a8ecbf7f083`, the unused Webview class has been removed, so this issue has been resolved.

Impact:

The WebView's settings expose attack surface unnecessarily.

Recommendation:

- If it's not required, consider disabling Javascript:
  `webview.configuration.preferences.javaScriptEnabled = false`
- Validate URLs to ensure the submitted string is a valid URL and uses HTTPS, and returns `nil` if not. For example a simple function like below can be added:

```
func validatedURL(from string: String) -> URL? {
        guard let url = URL(string: string), url.scheme == "https" else {
            return nil
        }
        return url
    }
}
```

- Use `.reloadIgnoringLocalCacheData` to ensure fresh content is always loaded.

2.3 HRZ-006 — Android: Confused GitHub configuration

Vulnerability ID: HRZ-006 **Status:** Unresolved

Vulnerability type: Development practices

Threat level: Low

Description:

The branch structure, release tags, and readme offered on GitHub are confusing and outdated, making it hard to identify and install the most up-to-date version.

Technical description:

The structure on GitHub makes it quite confusing and error-prone to get the newest app and code version.

At the time of writing, the master branch is somewhere around 2.0.4, and the last tagged release is 2.6.0. The master branch also contains a link to a 1.6.2 apk when clicking on "from this folder, as an APK, to be installed manually".

The develop branch is 2.7.1, which is the current version at the time of writing. The Play store also features 2.7.1. The develop branch also contains a link to a 1.6.2 apk when clicking on "from this folder, as an APK, to be installed manually".

For the FOSS version it is unclear to which branch is up to date. The release is up to date and matches the version on f-droid. The branches also contains a link to a 1.6.2 apk when clicking on "from this folder, as an APK, to be installed manually", which is outdated and is not the FOSS version.

Update :

Retest Result

Release 2.11.0 `https://github.com/Horizontal-org/Tella-Android/releases/tag/v2.11.0(186)` brings the master branch up to date with the Play Store. It replaced the download link in the readme with an up-to-date version. This resolves this situation for non-FOSS Tella. For Tella FOSS the situation remains unchanged: it is still outdated and the readme links to an outdated non-FOSS version.

Impact:

Users might install an old version or use old code.

Recommendation:

- Keep the main branches up to date. Update the links in the READMEs. Keep the releases up to date.

2.4 HRZ-009 — Android: Use of PBKDF2

Vulnerability ID: HRZ-009	**Status:** New
Vulnerability type: Cryptography configuration	
Threat level: Low	

Description:

PBKDF2 has limited brute-force resistance against GPUs, and iteration counts used are below recommendations.

Technical description:

The algorithm used to derive the main key from user input uses PBKDF2, which unfortunately is rather cheap to brute force using GPUs. Consequently, we recommend replacing it. See https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html for recommendations.

There is discussion about whether PBKDF2 depends on its parameters remaining within reach of government authorities to brute force: https://kolektiva.social/@cedar/110214532879538171 For Tella's target audience, we recommend switching to a more secure algorithm. Also, the iteration count is set to 10,000 is quite low, and below recommendations.

The file encryption also uses PBKDF2 with only 1,000 iterations. However, given that in this case the input is a full AES key and not something a human needs to remember, it is not realistic to brute force it.

In mitigation, to be able to mount a brute-force attack, an attacker would first have to extract the device data and therefore e.g root the phone.

Impact:

Brute-force difficulty is limited.

Recommendation:

- Increase PBKDF2 iteration counts.
- Change the key derivation algorithm to one that's more brute-force resistant (such as Argon2id).

2.5 HRZ-010 — Android: Manifest allows clear-text traffic

Vulnerability ID: HRZ-010	**Status:** Resolved
Vulnerability type: Cryptography configuration	
Threat level: Low	

Description:

The app manifest explicitly disables a protection mechanism against clear-text traffic.

Technical description:

The Android manifest contains

```
android:usesCleartextTraffic="true"
```

See https://developer.android.com/guide/topics/manifest/application-element#usesCleartextTraffic.

This flag explicitly instructs the underlying libraries that clear-text traffic is not forbidden. We have not observed the app sending clear-text traffic either way.

We suggest setting this flag to false, and making changes that required this setting in the first place to make it less likely that the app will actually send clear-text traffic.

Update :

Retest Result

As per the update done in `https://github.com/Horizontal-org/Tella-Android/pull/295/commits/8891f04b622fc5518bdbe0598614ac1fd4f0bd22`, this issue has been resolved.

Impact:

The app is not prevented from making unencrypted connections, potentially exposing traffic to interception.

Recommendation:

- Do not allow clear-text traffic in Android manifest.

2.6 HRZ-007 — Android: Outdated 3rd-party dependencies

Vulnerability ID: HRZ-007 **Status:** Unresolved

Vulnerability type: Outdated dependencies

Threat level: Unknown

Description:

Tella Android uses dozens of outdated or deprecated dependencies, some of which contain known vulnerabilities.

Technical description:

Tella Android uses dozens of outdated dependencies, several of which contain vulnerabilities. We did not further analyze which have an actual impact on the product, as we recommend to always keep product security up to date. Spot checks in the list of known vulnerabilities did not reveal anything exploitable.

We recommend running https://github.com/dependency-check/dependency-check-gradle or other automatic analyzers to get an overview; we used that one for our test too.

However, this tool will not show deprecated/abandoned libraries. One essential security library https://github.com/sqlcipher/android-database-sqlcipher is deprecated and should be replaced, even though here are currently no known vulnerabilities in this dependency.

Android FOSS dependencies are even more outdated, because it is based on an old version of Tella Android.

Update :

Retest Result

Release 2.11.0 `https://github.com/Horizontal-org/Tella-Android/releases/tag/v2.11.0(186)` replaces `android-database-sqlcipher` with an up-to-date and maintained library `sqlcipher-android`. However, dozens of known vulnerable libraries remain in the product. Spot checks in the list of known vulnerabilities did not reveal anything exploitable, but assessing the impact of them all exceeds the time available for this review. We still recommend keeping all dependencies up to date.

Impact:

Unknown, vulnerable dependencies might impact Tella app security.

Recommendation:

- Update outdated dependencies.
- Check and replace deprecated dependencies.

3 Non-Findings

In this section we list some of the observations that did not directly result in a vulnerability.s.

3.1 NF-001 — Android: Tella FOSS is outdated

We compared Tella FOSS to the normal Tella

First we confirmed that the statement in the change log is correct: https://tella-app.org/releases/

> Android: Tella FOSS 2.0.15 (based on Android 2.0.15) - released on July 10, 2023 A version of Tella included for the first time on the F-droid store. This a 100% Free and Open-Source Software (FOSS) version of Tella Android. We removed all trackers, changed map and location provider and also changed the Camera library to CameraX, removed crashlytics, LoggingInterceptor and any other non-FOSS component or dependency. We removed completely all Google Play Services dependencies.

For that we compared Tella 2.0.12 https://github.com/Horizontal-org/Tella-Android/commit/739e1ea055fc9735aab2b54a4c58611f0038ea0f to https://github.com/Horizontal-org/Tella-Android-FOSS/releases/tag/v2.0.15(149). We did not identify a normal Tella 2.0.15 commit so used 2.0.12 instead.

However, Tella FOSS is quite outdated and lacks security features such as "Restrict unlocking attempts", bug fixes, and other features. This is also mentioned in the feature comparison https://tella-app.org/features/.

Tella FOSS does not contain security fixes that have been applied to the non-FOSS version.

3.2 NF-002 — Android: Camouflage

The observations related to "Android App Camouflage" from the subgraph still applies. The camouflage works great against casual inspection and the app cannot be found in the launcher and e.g. behaves like a normal calculator. Deeper inspection e.g. by looking into the settings menu still reveals that Tella is installed. In theory this can be resolved by making other apps that fully imitate a calculator app without ever mentioning Tella which can be installed instead of the un-camouflaged Tella app. However, adversaries would then just need to look for this specific app, and would also know it is actually Tella, resulting in a never-ending cat and mouse game.

3.3 NF-003 — Android: Subgraph Finding V-001 resolved

We consider Subgraph's finding "V-001 Unrestricted Unlock Attempts" resolved, because the app now allows setting a limit on the number of unlock attempts before data is permanently deleted.

This is not implemented in Tella-FOSS.

3.4 NF-004 — Android: Unlock method security

Subgraph's notes on Android "Application Unlocking" still apply.

The UI is overselling when stating PIN security is moderately secure. Depending on the adversary, a 6 digit pin with the chosen algorithm (see HRZ-009 (page 12)) is not brute-force resistant. This provides almost no security compared to a strong password. However, to be able to mount a brute-force attack an attacker would first have to extract the device data and therefore e.g root the phone.

3.5 NF-005 — Android: CTR mode

CTR mode, the block cipher mode of operation which is used for file encryption is malleable. A bit-flip in the ciphertext results in a bit-flip in the plaintext. This can e.g. be used to manipulate files headers. Given that Tella's security promise is encryption and not integrity protection, this seems fine, just a limitation to be aware of. Choosing a different block cipher mode of operation such as AES-GCM improves security by providing both encryption and integrity protection.

3.6 NF-008 — Android: Exposure through analytics

Observations on Android Analytics and Crash Data Collection from Subgraph's report still apply. The crash analytics are opt-out rather than opt-in.

This is resolved in the FOSS version.

4 Future Work

- **Retest of findings**
  When mitigations for the vulnerabilities described in this report have been deployed, a repeat test should be performed to ensure that they are effective and have not introduced other security problems.
- **Regular security assessments**
  Security is an ongoing process and not a product, so we advise undertaking regular security assessments and penetration tests, ideally prior to every major release or every quarter.

5 Conclusion

We discovered 2 Moderate, 3 Low and 1 Unknown-severity issues during this penetration test.

We recommend fixing all of the issues found and then performing a retest in order to ensure that mitigations are effective and that no new vulnerabilities have been introduced.

Finally, we want to emphasize that security is a process – this penetration test is just a one-time snapshot. Security posture must be continuously evaluated and improved. Regular audits and ongoing improvements are essential in order to maintain control of your corporate information security. We hope that this pentest report (and the detailed explanations of our findings) will contribute meaningfully towards that end.

Please don't hesitate to let us know if you have any further questions, or need further clarification on anything in this report.

Appendix 1 Testing team

Tim Hummel	Tim Hummel is a Security Expert and developer. His specialty is hardware, crypto, and related software security. In his work he tests everything from apps, car components, payment solutions, white-box crypto, pay TV, smart cards, mobile devices, IoT, TPMs, TEEs, bootloaders, entertainment systems, transport cards to web services and APIs.
Niko Schmidt	Niko has worked as an IT Security Analyst for several companies. He holds a Master in IT Security. During his studies he started a CTF contest and is a co-founder of the ALLES! CTF team.
Abhinav Mishra	With over a decade of experience, Abhinav is a seasoned professional specializing in penetration testing for web, mobile, and infrastructure systems. Additionally, Abhinav is highly regarded for his expertise in delivering comprehensive training programs focused on enhancing security measures for web, mobile, and infrastructure technologies.
Melanie Rieback	Melanie Rieback is a former Asst. Prof. of Computer Science from the VU, who is also the co-founder/CEO of Radically Open Security.